Trabajo práctico final

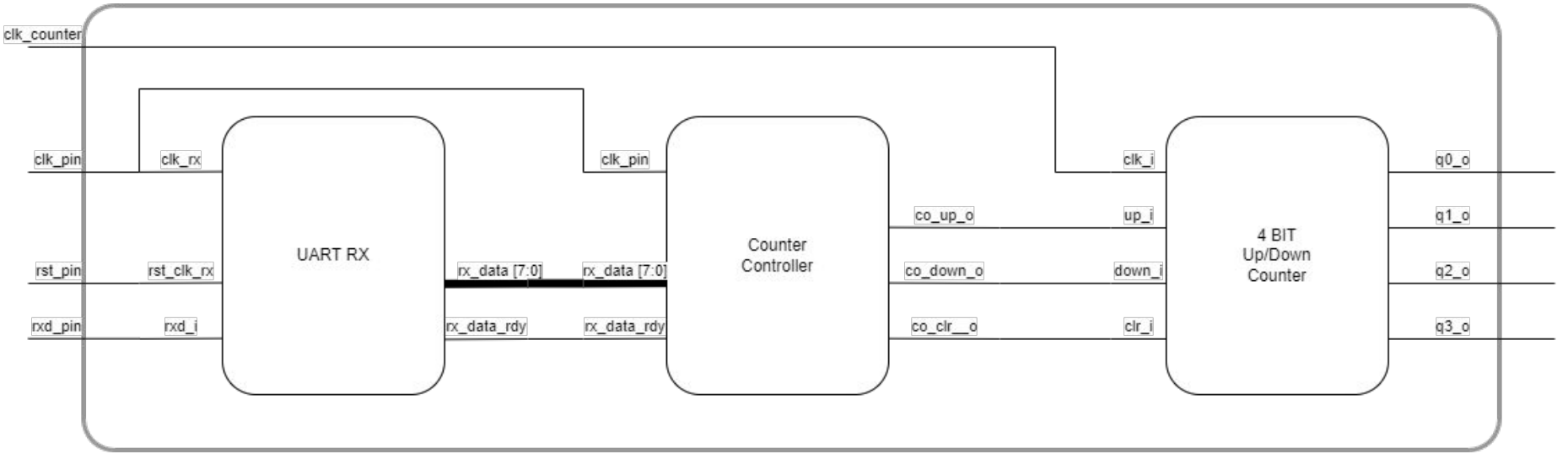
Circuitos Lógicos Programables

Especialización en Sistemas Embebidos

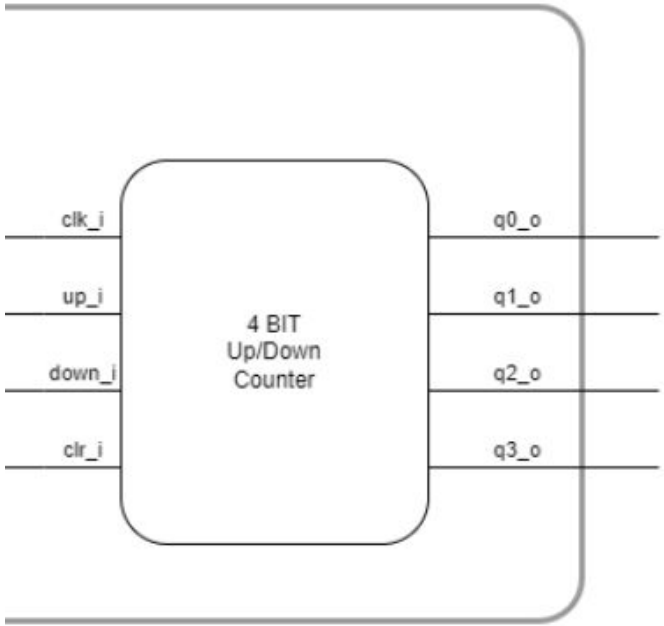
Fernando Nicolas Calvet (fernando.n.calvet@gmail.com)

Diagrama general del hardware a implementar

El módulo UART RX recibe datos serie, el controlador del contador interpreta los comandos y el contador ascendente/descendente de 4 bits incrementa, decrementa o reinicia según los comandos.

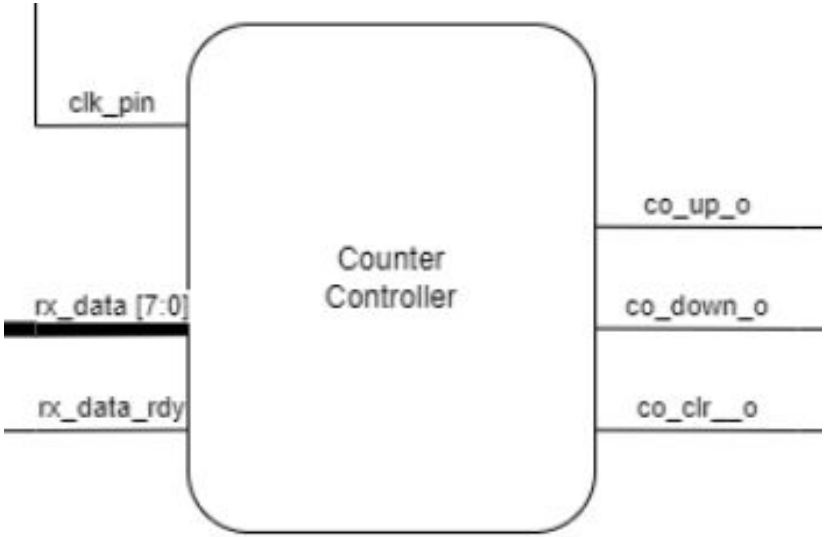


Modulo contador



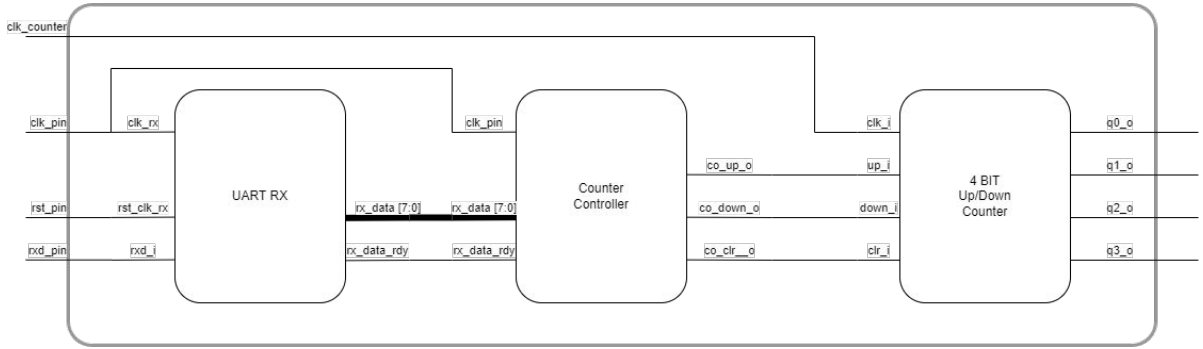

```
6   ENTITY contUpDw IS
7       PORT (
8           clk_i : IN STD_LOGIC;
9           clr_i : IN STD_LOGIC;
10          up_i :  IN STD_LOGIC;
11          down_i :IN STD_LOGIC;
12          count : OUT STD_LOGIC_VECTOR (3 DOWNTO 0)
13          );
14  END contUpDw;
15
16  ARCHITECTURE contUpDw_arq OF contUpDw IS
17
18      SIGNAL aux:STD_LOGIC_VECTOR(3 DOWNTO 0) := "0000";
19
20  BEGIN
21      PROCESS (clk_i, clr_i)
22      BEGIN
23
24          IF (clr_i = '1') THEN
25              aux <= "0000";
26          ELSIF (rising_edge(clk_i)) THEN
27              IF (up_i = '1') THEN
28                  aux <= aux + 1;
29              ELSif(down_i = '1') THEN
30                  aux <= aux - 1;
31              END IF;
32          END IF;
33      END PROCESS;
34
35      count <= aux;
36  END contUpDw_arq;
```

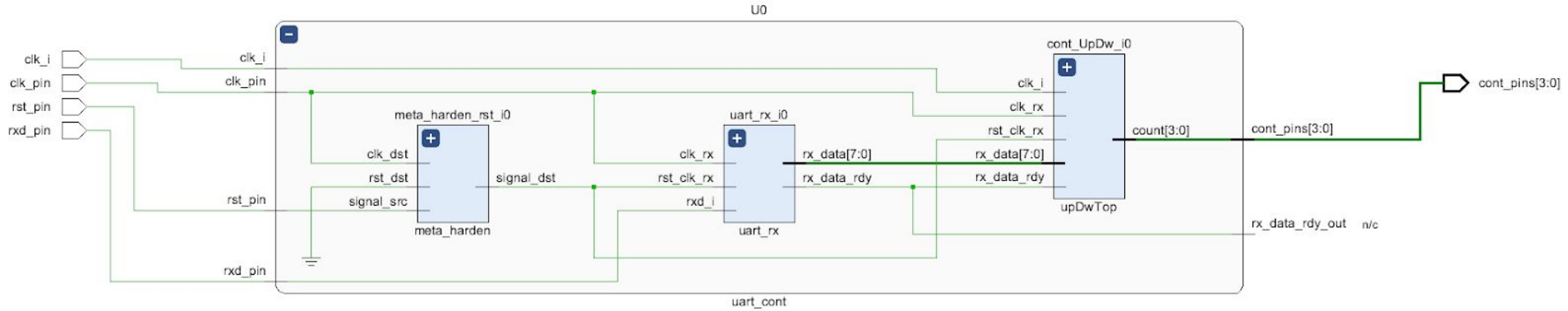
Modulo controlador



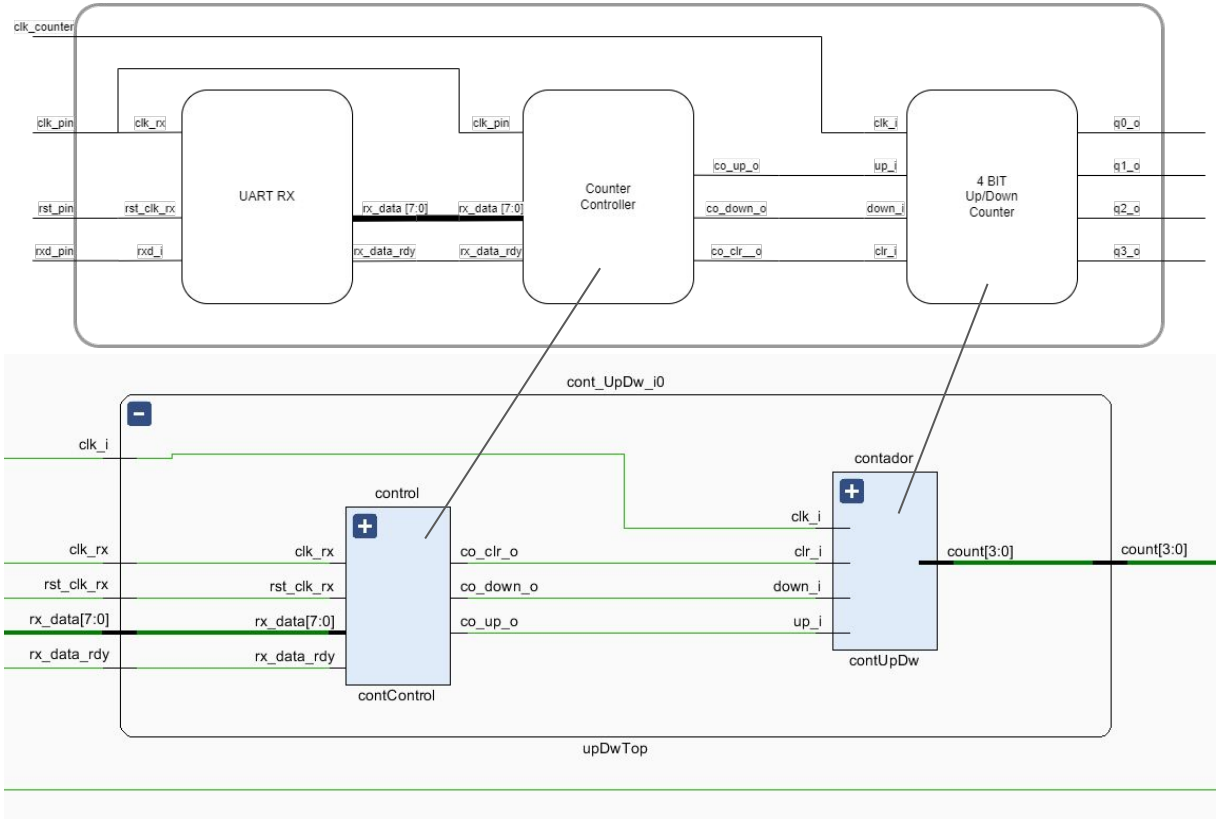

```
ENTITY contControl IS
    PORT (
        clk_rx : IN STD_LOGIC; -- Clock input
        rst_clk_rx : IN STD_LOGIC; -- Active HIGH reset - synchronous
        rx_data : IN STD_LOGIC_VECTOR(7 DOWNTO 0);-- 8 bit data output
        rx_data_rdy : IN STD_LOGIC; -- valid when rx_data_rdy is asser
        co_up_o : OUT STD_LOGIC;
        co_down_o : OUT STD_LOGIC;
        co_clr_o : OUT STD_LOGIC
    );
END;

ARCHITECTURE contControl_arq OF contControl IS
    SIGNAL old_rx_data_rdy : STD_LOGIC;
    SIGNAL char_data : STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN

    PROCESS (clk_rx)
    BEGIN
        IF rising_edge(clk_rx) THEN
            IF rst_clk_rx = '1' THEN
                old_rx_data_rdy <= '0';
                char_data <= "00000000";
            ELSE

                old_rx_data_rdy <= rx_data_rdy;

                IF (rx_data_rdy = '1' AND old_rx_data_rdy = '0') THEN
                    char_data <= rx_data;

                    IF (rx_data = "01010101") THEN --U
                        co_up_o <= '1';
                        co_down_o <= '0';
                    END IF;
                    IF (rx_data = "01000100") THEN --D
                        co_up_o <= '0';
                        co_down_o <= '1';
                    END IF;
                    IF (rx_data = "01000011") THEN --C
                        co_clr_o <= '1';
                        co_up_o <= '0';
                        co_down_o <= '0';
                    END IF;
                else
                    co_clr_o <= '0';
                END IF;
            END IF; -- if !rst
        END IF;
    END PROCESS;
END;
```

RTL analysis

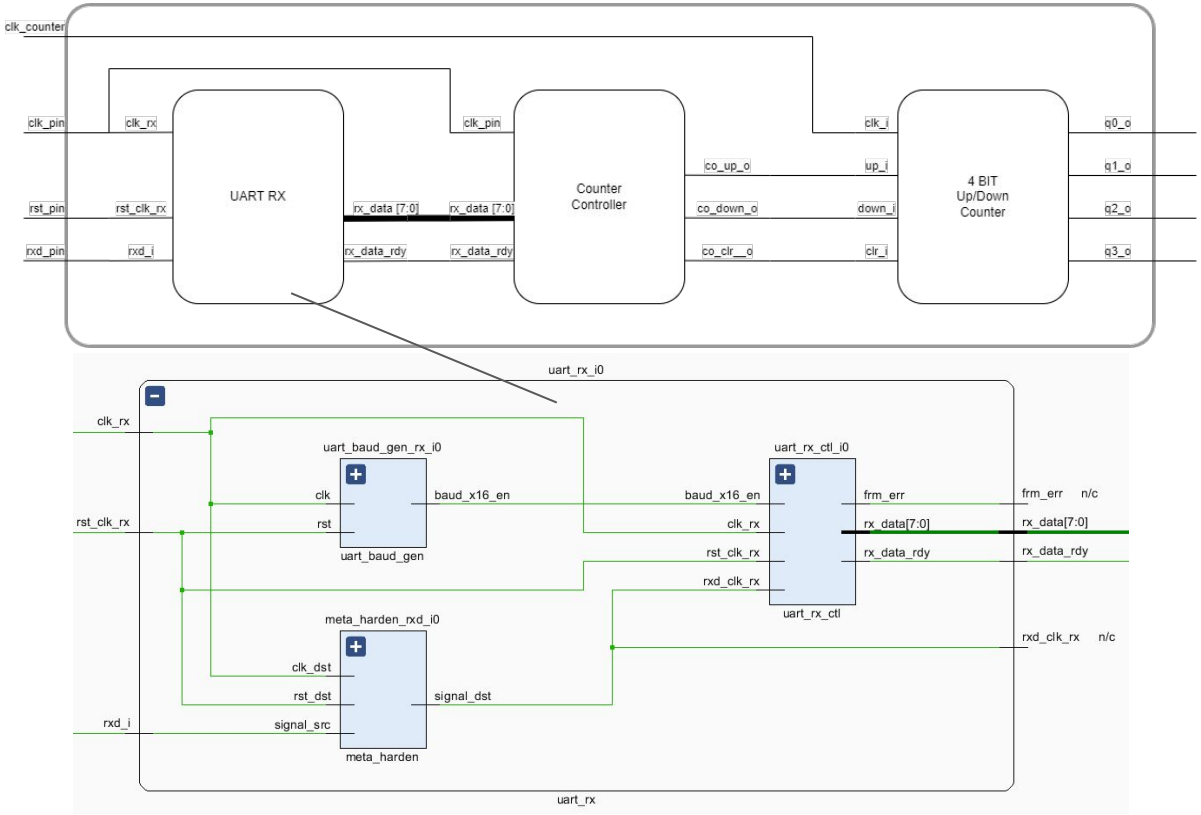




RTL analysis

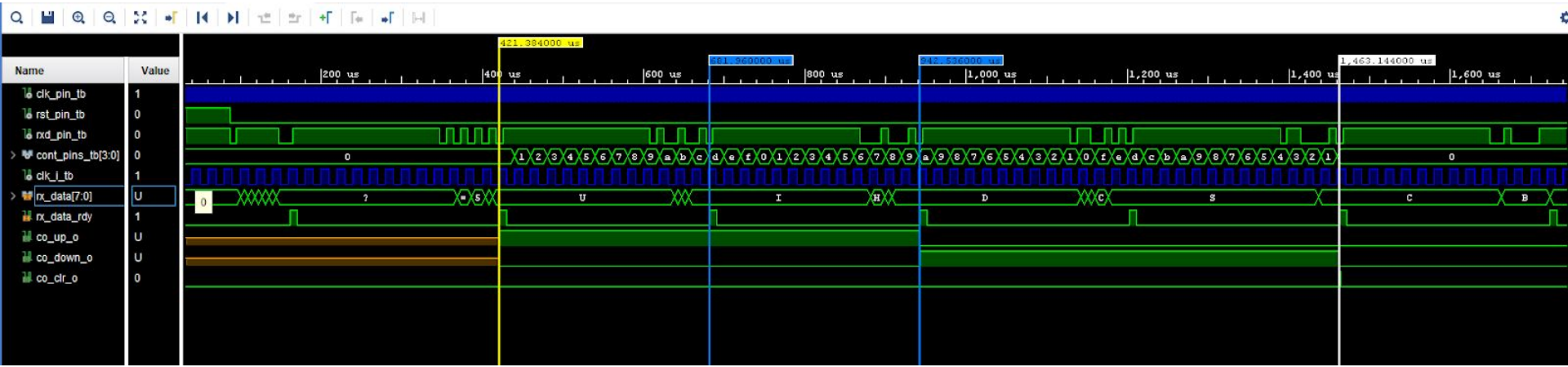


RTL analysis



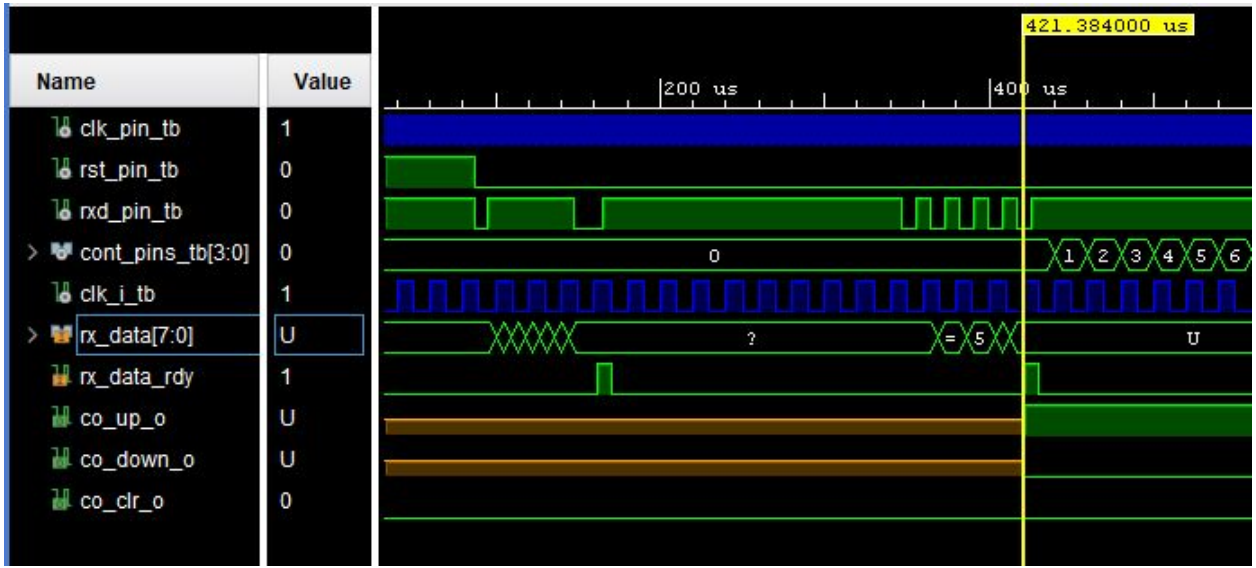
Simulacion

Completo



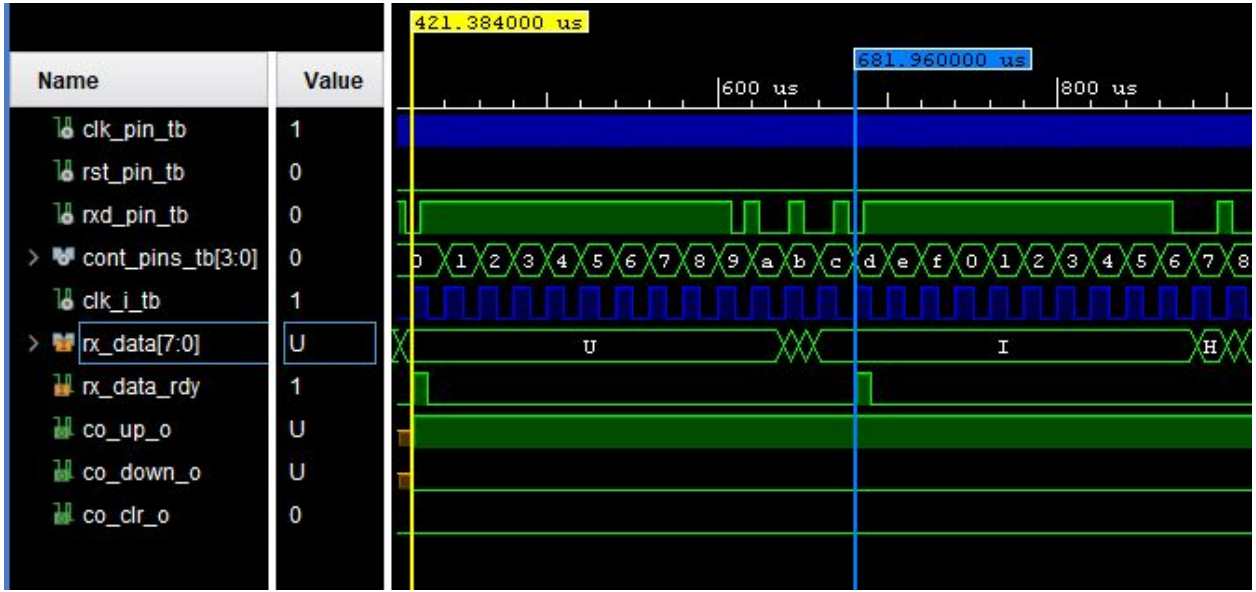
Simulacion

Marcador 1



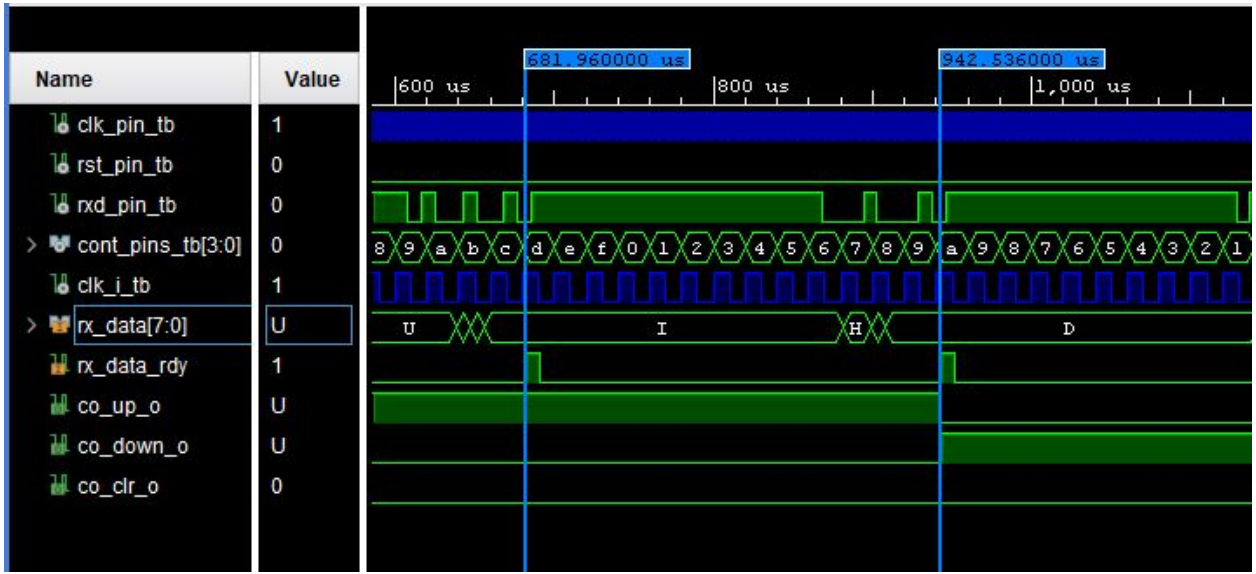
Simulacion

Marcador 2



Simulacion

Marcador 3



Simulacion

Marcador 4

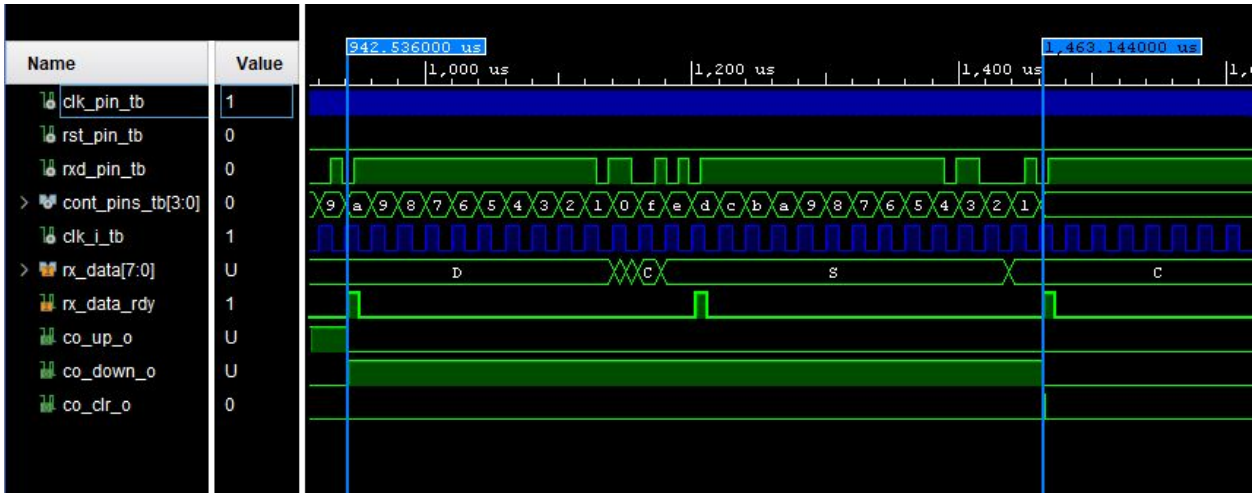
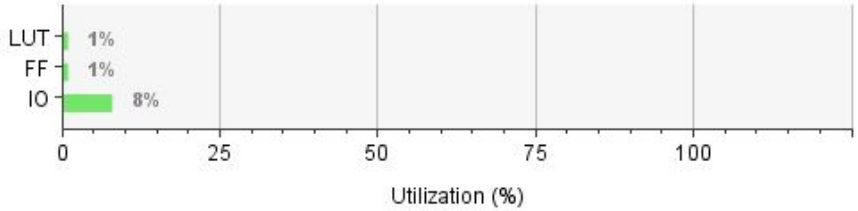


Tabla de uso de recursos

Resource	Utilization	Available	Utilization %
LUT	46	17600	0.26
FF	40	35200	0.11
IO	8	100	8.00



Name	Slice LUTs (17600)	Slice Registers (35200)	Bonded IOB (100)	BUFGCTRL (32)
uart_top	46	40	8	2
U0 (uart_cont)	46	40	0	0
cont_UpDw_i0 (up...	3	8	0	0
meta_harden_rst_i...	0	2	0	0
uart_rx_i0 (uart_rx)	43	30	0	0
meta_harden_rx...	0	2	0	0
uart_baud_gen_...	8	8	0	0
uart_rx_ctl_i0 (ua...	35	20	0	0

Posibilidades de mejora de implementación

Sería interesante realizar el contador de forma configurable tal que pueda utilizarse un “generic” para indicarle los bits de salida.
También podría mejorarse la implementación del testbench, ya que no se utiliza ninguna optimización al momento de enviar los datos simulados por la puerta serie.